2023012843 李宛诺

一.名词解释

1.操作系统：操作系统是配置在计算机硬件上的第一层软件，是对硬件系统的首次扩充，其主要作用是管理硬件设备，提高它们的利用率和系统吞吐量，并为用户和应用程序提供一个简单的接口，以便于用户和应用程序使用硬件设备。

4.实时操作系统：实时系统是指系统能及时响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行。

5.互斥共享：当进程A要访问某资源必须提出请求，若此时该资源空闲，系统便可将其分配给进程A。此后再有其他进程也要访问时只要A未用完，其他进程就必须等待。仅当A对该资源访问完并释放后才允许另一进程对该资源进行访问。

三.1.硬件系统 软件系统 3.硬件资源 软件资源 5.共享性互斥性

一.名词解释

3.临界区：指进程中访问临界资源的那段代码区域。临界资源是一次仅允许一个进程访问的资源。多个进程进程如果同时进入临界区访问临界资源，可能会导致数据不一致，结果错误等问题

4.进程同步：是指对多个相关进程在执行次序上进行协调，使这些进程在某些关键点上按照预定的顺序执行，以保证系统中多个进程能有序地共享资源和相互合作。

三.1.5.动态性.异步性并发性 5.动态.静态 6.进程同步

8.共享存储器；消息传递管理通信 9.就绪：执行.阻塞

五、2. $S_1 \to S_2 \to S_4 \to S_6 \to S_7$

$S_1 \to S_3 \to S_5 \to S_6 \to S_7$

3. semaphore a=0, b=0, c=0, d=0, f=0, g=0

process S_1(){ v(a); v(b);}

process S_2(){ p(a); v(c);}

process S_3(){ p(b); v(d);}

process S_4(){ p(c); v(e);}

process S_5(){ p(d); v(f);}

process S_6(){ p(e); p(f); v(g);}

process S_7(){ p(g);}

---

3. semaphor mutex=1;
semaphore empty=5;
Semaphore apple=0;
Semaphore orange=0;

```
void father(){
  while(true){
  p(empty);
  p(mutex);
  if(随机选苹果){v(apple);}
  else{v(orange);}
   v(mutex);}}
```

```
void son{
   while(true){
   p(orange); p(mutex);
   v(empty); v(mutex);
        }}
```

```
void daughter(){
  while(true){
  p(apple);
  p(mutex);
  v(empty);
  v(mutex);
  }}
```

2023012843 李晓洁

4.
```
semaphore empty1=0;
semaphore full1=0
semaphore empty2=1
semaphore full2=0
void PA(){
while (true){
P(empty1)
V(full1); }}
void PB(){
while (true){
P(full1);
P(empty2);
V(empty1);
V(full2); }}
void PC(){
while (true){
P(full2);
V(empty2); }}
```

5.
```
semaphore customers=0
semaphore baber=0
semaphore mutex=1
semaphore payment=0
semaphore receipt=0
int waiting=0 Const int N=5
void baber(){
while (true){
P(customers);
P(mutex);
waiting--;
V(barber); V(mutex);
P(payment) V(receipt);}}
Void customer(){
P(mutex);
if(waiting<N){
waiting++;
V(customers);
V(mutex);
P(barber);
V(payment);
P(receipt);}
else{ V(mutex);}}
```

一、名词解释.

2. 处理机调度：是对处理机进行分配。处理机调度算法是指根据处理机分配策略所规定的处理机分配算法。在多道批处理系统中，一个作业从提交到获得处理机执行直至作业运行完毕，可能需要经历多级处理机调度。

3. 周转时间：是指一个进程从提交到系统开始，直到最终完成所经历的总时间。它包括进程在就绪队列中的等待时间、CPU执行时间以及I/O操作等其他所有的时间。

4. 死锁：指两个或多个进程因竞争资源而陷入互相等待状态，若无外力干预，所有进程都将无法向前推进。

二、

2. 提交后备完成 b. 预防死锁 避免死锁 检测死锁 解除死锁

三、1. ①先来先服务FCFS

执行顺序 ABCDE

完成时间：A：3 时刻 B：3+6=9

C 9+4=13 D：13+5=18 E：18+2=20

周转时间：A：3-0=3 B：9-2=7

C：13-4=9 D：18-6=12 E：20-8=12

带权周转时间：A：3/3=1 B：7/6≈1.17

C：9/4≈2.25 D：12/5=2.4 E 12/2=6

平均周转时间：(3+7+9+12+12)/5=8.2

平均带权周转时间：(1+1.17+2.25+2.4+6)/5

=2.564

②. 非抢占短作业优先(SJF)

执行顺序：A C B E D

完成时间 A：3 C：3+4=7

B 7+6=13 E：13+2=15 D：15+5=20

周转时间：A：3-0=3 C=7-4=3

B：13-2=11 E 15-8=7 D：20-6=14

带权周转时间 A：3/3=1 C 3/4=

B：11/6≈1.83 E：7/2=3.5 0.75

D：14/5=2.8

平均周转时间：

(3+3+11+7+14)/5=7.6

平均带权周转：(1+0.75+1.83+3.5+2.8)/5=

1.976

③抢占短作业优先(SJF)

执行顺序：ACBED

完成时间：A:3. C:3+4=7

B:7+5=12 E:12+2=14 D:14+5=19

周转时间：A:3-0=3. C:7-4=3.

B:12-2=10 E:14-8=6. D:19-6=13

带权周转时间：

A:3/3=1 C:3/4=0.75 B:10/6=1.67

E:6/3=3 D:13/5=2.6

平均周转时间：

(3+3+10+6+13)/5=7

平均带权周转时间：

(1+0.75+1.67+3+2.6)/5=1.804

④高响应比优先(HRRN)

执行过程 ABCEGD

完成时间：A:3. B:3+6=9 C:9

+4=13. E:13+2=15. D:15+5=20

周转时间 A:3-0=3.

B:9-2=7 C:13-4=9 E:15-8=7

D:20-6=14

带权周转时间：

A:3/3=1 B:7/6=1.17 C:

9/4=2.25 E:7/2=3.5

D:14/5=2.8

平均周转:(3+7+9+7+14)/5=8

平均带权:(1+1.17+2.25+3.5

+2.8)/5=2.144

⑤时间片轮转：

执行过程 A B C D E -

完成时间 A:4 B:13 C:9 D:18 E:14

周转时间：A:4-0=4 B:13-2=11 C:9-4=5 D:18-6=12

E:14-8=6

带权周转时间：A:4/3=1.33 B:11/6=1.83 C:5/4=1.25

D:12/5=2.4 E:6/2=3

平均:(4+11+5+12+6)/5=7.6

平均带权(1.33+1.83+1.25+2.4+3)/5=1.962

2.

任务A截止时间为 20ms.

松弛度：20-0-10=10.

任务B截止：50ms

松弛度：50-0-10=40

任务C截止：50ms

松弛度：50-0-25=25

A松弛度最小执行A.

A执行完：B松弛度：50-10-10=30

C：50-10-25=25.

C本松弛度最小，执行C

C执行完，任务B：50-25-10=15.

执行B.

之后按周期重复ACB

3. 1. 计算：分配矩阵：

Allocation；Max-Need：P0：0 0 3 2；

P1：1 0 0 0；P2：1 3 5 4；P3：0 6 3 2；

P4：0 0 1 4

安全性检查：Work=Available=1 6 2 2.

P0满足。Work=Work+Allocation[0]=1 6 5 4.

P3满足 Work=1 9 8 6；P4满足。Work=1 9 9 10，

P1满足。Work：2 9 9 10。P2满足。存在安全序列<P0，P4，P3，P1，P2>，

2. 判断能否分配给P2.

Request[1,2,2,2]≤Need[2]且Request[1,2,2,2]≤Available.

可以分配。

分配后 Available变为0 4 3 0

Need[2]变成 1 1 3 4

Allocation[2]变为 2 5 7 6

3. 判断死锁

不会。因为分配系统

假如存在有安全序列

可使进程顺利执行完成